

```
# AWS-Load-balancer  
Launch EC2 Instance (Web Server – 1)  
Main Task : Distributing Traffic Between Two Apache Web Servers Using AWS Application Load Balancer
```

Task 1: Create EC2 Instance

Logged in to AWS Management Console

Navigated to EC2 → Instances → Launch Instance

Instance Name: web-server-1

AMI: Amazon Linux 2

Instance Type: t2.micro

Key Pair: Selected existing key pair

Security Group Configuration

Created a new security group with the following inbound rules:

Type	Protocol	Port	Source
HTTP	TCP	80	Anywhere (0.0.0.0/0) (Reason: HTTP access is required for web)
SSH	TCP	22	(default)

Instance state = running		X	Clear filters			
<input type="checkbox"/>	Name ⚖	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	webserver1	i-04d1875c6a871c7d9	Running	t2.micro	Initializing	View alarms +

Launch Instance

Task 2: Launch Second EC2 Instance (Web Server – 2)

Create EC2 Instance

- Navigated to **EC2 → Instances → Launch Instance**
- Instance Name: **web-server-2**
- AMI: **Amazon Linux 2**
- Instance Type: **t2.micro**
- Key Pair: Selected existing key pair

Configure Network Settings

Security Group Configuration

Created a new security group with the following inbound rules:

Type	Protocol	Port	Source
HTTP	TCP	80	Anywhere (0.0.0.0/0) (Reason: HTTP access is required for web)
SSH	TCP	22	(default)

Instance state = running		Clear filters						
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Pub
<input type="checkbox"/>	webserver2	i-0c837efb77d81f62e	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1b	ec2-
<input type="checkbox"/>	webserver1	i-04d1875c6a871c7d9	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1b	ec2-

Task 3: Connecting EC2 Instances to Local Terminal –webserver1

Steps: terminal

>>Navigate to Downloads Folder

cd ~/Downloads

>>Set Key File Permissions

ls

>>Connect to web-server-1

ssh -i my-key.pem ec2-user@<web-server-1-public-ip>

>>Switch to Root User

Sudo -i

Task 4: To install Apache HTTP Server

sudo yum install httpd -y

>>To start the web server

sudo systemctl start httpd

sudo systemctl enable httpd

>>check the status

systemctl status httpd

```
[root@ip-172-31-13-166 ~]# service httpd start
Redirecting to /bin/systemctl start httpd.service
[root@ip-172-31-13-166 ~]# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Active: active (running) since Mon 2025-12-15 14:12:37 UTC; 13s ago
       Docs: man:httpd.service(8)
   Main PID: 27271 (httpd)
      Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0"
     Tasks: 177 (limit: 1120)
    Memory: 13.0M
      CPU: 70ms
     CGroup: /system.slice/httpd.service
             └─27271 /usr/sbin/httpd -DFOREGROUND
                  ├─27295 /usr/sbin/httpd -DFOREGROUND
                  ├─27298 /usr/sbin/httpd -DFOREGROUND
                  ├─27299 /usr/sbin/httpd -DFOREGROUND
                  └─27300 /usr/sbin/httpd -DFOREGROUND

Dec 15 14:12:37 ip-172-31-13-166.ap-south-1.compute.internal systemd[1]: Starting httpd.service .
Dec 15 14:12:37 ip-172-31-13-166.ap-south-1.compute.internal systemd[1]: Started httpd service .

To exit from this press ctrl+c
```

>>once exit from this status it will come to ec2 user again change the user to root user

Cmd : sudo -i

Task 5: Create HTML File

>>Navigate to Web Directory

cd /var/www/html

>>Create HTML File

vi index.html

Sample code

```
<!DOCTYPE html>
<html>
<head>
  <title>AWS Load Balancer Demo</title>
  <style>
    body {
      margin: 0;
      font-family: Arial, Helvetica, sans-serif;
      background: linear-gradient(to right, #141e30, #243b55);
      color: white;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
    }
    .card {
      background: #1f2a44;
      padding: 30px;
      border-radius: 12px;
      text-align: center;
```

```

        box-shadow: 0 10px 25px rgba(0,0,0,0.4);
        width: 400px;
    }

    h1 {
        margin-bottom: 10px;
        color: #4fd1c5;
    }

    p {
        font-size: 16px;
        margin: 8px 0;
    }

    .server {
        font-size: 18px;
        font-weight: bold;
        color: #f6e05e;
    }

```

</style>

</head>

<body>

```

<div class="card">
    <h1>AWS Load Balancer</h1>
    <p>Web Application is Running Successfully 🚀</p>
    <p class="server">Server Name: WEB-SERVER-1</p>
    <p>Application Load Balancer Demo</p>
</div>

```

</body>

</html>


```

>>Exit Editor
:q!
>> restart the service
service httpd restart
[ec2-user@ip-172-31-13-166 html]$ sudo -i
[root@ip-172-31-13-166 ~]# cd /var/www/html
[root@ip-172-31-13-166 html]# vim index.html
[root@ip-172-31-13-166 html]# service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-13-166 html]#

```

>>go to chrome copy the public ip address and add http port
Ex: 13.203.75.16:80
Olp :web page



Task 6: Connecting EC2 Instances to Local Terminal –webserver2

>>repeat the same steps which we did in webserver1
>>Switched to root user.
>>Installed Apache web server.
sudo yum install httpd -y
>>Started and enabled Apache service.
sudo systemctl start httpd
sudo systemctl enable httpd
>>Verified Apache service status.
systemctl status httpd

Task 7: Created an HTML file.

```
cd /var/www/html
vi index.html
```

Sample code for web server -2

```
<!DOCTYPE html>
<html>
<head>
    <title>AWS Load Balancer - Server 2</title>
    <style>
        body {
            margin: 0;
            font-family: Arial, sans-serif;
        }
    </style>
</head>
<body>
    <h1>Welcome to AWS Load Balancer - Server 2</h1>
    <p>This is a sample web page served by the AWS Load Balancer.</p>
    <ul>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
    </ul>
</body>
</html>
```

```
background: linear-gradient(to right, #0f2027, #203a43, #2c5364);
color: #ffffff;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
}

.box {
background: rgba(0, 0, 0, 0.6);
padding: 35px;
border-radius: 14px;
text-align: center;
box-shadow: 0 12px 30px rgba(0,0,0,0.5);
width: 420px;
}

h1 {
color: #ffb703;
margin-bottom: 12px;
}

p {
font-size: 16px;
margin: 8px 0;
}

.server {
font-size: 20px;
font-weight: bold;
color: #90dbf4;
}

</style>
</head>
<body>
<div class="box">
<h1>Application Load Balancer</h1>
<p>Web Application is running successfully ✓</p>
<p class="server">Server Name: WEB-SERVER-2</p>
<p>Traffic served by EC2 Instance 2</p>
</div>

</body>
</html>
```

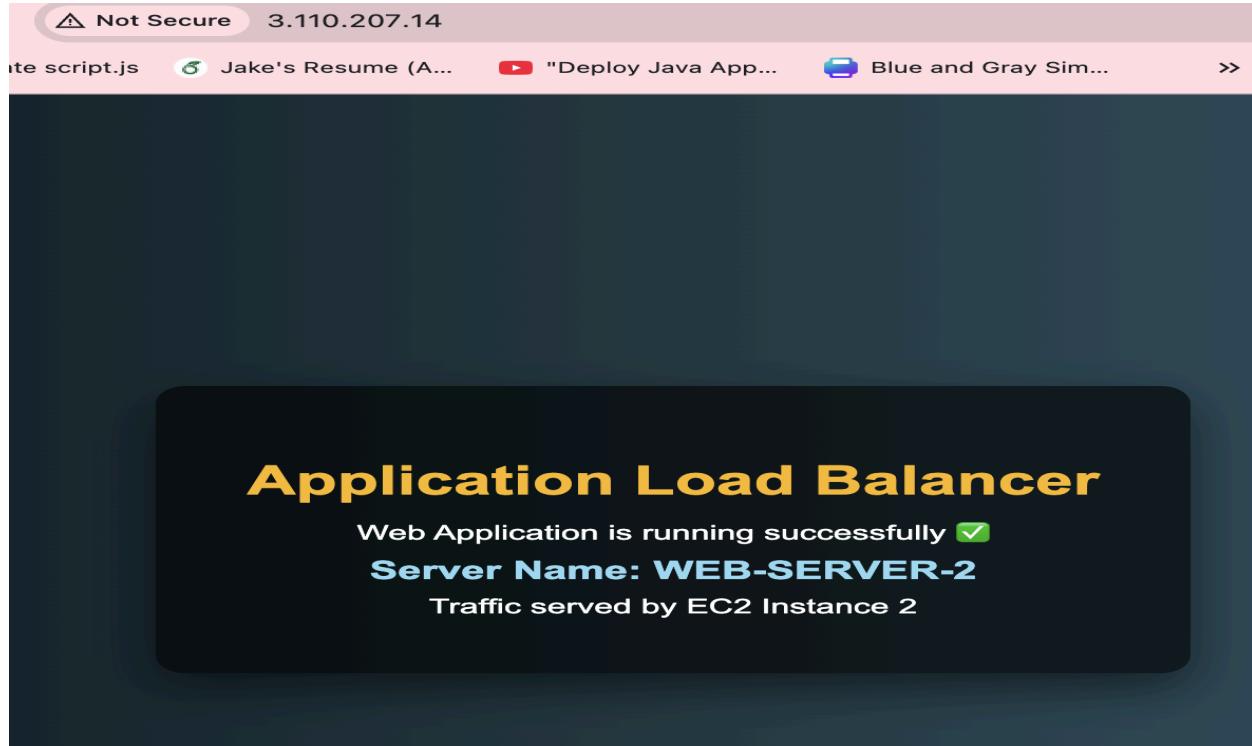
>>Exit Editor

:q!

>>go to chrome copy the public ip address and add http port

Ex: 3.110.207.14:80

Olp :web page



Info : In real time, we don't share server IP addresses because it is not safe.
use a load balancer to share traffic between servers so the application works fast

Task 8:Create Target group

Info : A target group connects the load balancer to the web servers so traffic goes to the right servers.

Process: Go to EC2 → Target Groups.

A screenshot of the AWS EC2 Target Groups page. The left sidebar shows navigation links like 'EC2', 'Target groups', 'Capacity Reservations', 'Capacity Manager', and sections for 'Images' and 'Elastic Block Store'. The main content area has a header 'Target groups' with 'Actions' and 'Create target group' buttons. It includes a search bar 'Filter target groups' and a table with columns for 'Name', 'ARN', and 'Port'. The table shows one item with the name 'my-target-group'.

>> configure the Target group

Enter the target name: target 1

Target type : instances

Create target group

A target group can be made up of one or more targets. Your load balancer routes requests to the targets in a target gr

Settings - immutable

Choose a target type and the load balancer and listener will route traffic to your target. These settings can't be mo

Target type

Indicate what resource type you want to target. Only the selected resource type can be registered to this target group.

Instances

Supports load balancing to instances in a VPC.
Integrate with Auto Scaling Groups or ECS services for automatic management.

Suitable for: **ALB** **NLB** **GWLB**

IP addresses

Supports load balancing to VPC and on-premises resources. Facilitates routing to IP addresses and network interfaces on the same instance. Supports IPv6 targets.

Suitable for: **ALB** **NLB** **GWLB**

Lambda function

Supports load balancing to a single Lambda function.
ALB required as traffic source.

Suitable for: **ALB**

Application Load Balancer

Allows use of static IP addresses and PrivateLink with an Application Load Balancer. NLB required as traffic source.

Suitable for: **NLB**

Target group name

Name must be unique per Region per AWS account.

target1

Next keep every thing by default click on **Next**

>>select the instances for target group

Available instances (2/2)

Filter instances				
<input checked="" type="checkbox"/>	Instance ID	Name	State	Security groups
<input checked="" type="checkbox"/>	i-0c837efb77d81f62e	webserver2	Running	launch-wizard-4
<input checked="" type="checkbox"/>	i-04d1875c6a871c7d9	webserver1	Running	launch-wizard-3

>>click on create a target group

Target groups (1)

[Info](#) | [What's new?](#)

Actions ▾

Create target group

Filter target groups

< 1 > |

<input type="checkbox"/>	Name	ARN	Port
<input type="checkbox"/>	target1	arn:aws:elasticloadbalancin...	80

Task 8:Create Load balancer

Load balancers [What's new?](#)



Actions ▾

Create load balancer



>>Select:Application load balancer

>> configure the Application load balancer

Load balancer name : applicationloadbalancer

>> Need to select atleast two availability zones to enroute the load balancer traffic

Availability Zones and subnets | [Info](#)

Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each zone only.

ap-south-1a (aps1-az1)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses per zone.

subnet-0c8733dc2601e2b88

IPv4 subnet CIDR: 172.31.32.0/20

ap-south-1b (aps1-az3)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses per zone.

subnet-0f267fd671d9f0c20

IPv4 subnet CIDR: 172.31.0.0/20

ap-south-1c (aps1-az2)

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses per zone.

subnet-0836785c0f40836ac

IPv4 subnet CIDR: 172.31.16.0/20

>>create security group. I created security group name it has HTTP - IPV4

⌚ Security group (sg-080dbce6964d40421 | HTTP) was created successfully
▶ Details



sg-080dbce6964d40421 - HTTP

Actions ▾

Details

Security group name



HTTP

Security group ID



sg-080dbce6964d40421

Description



Allow

VPC ID

vpc-03e487a9440b8ff6b ↗

Owner

801458814717

Inbound rules count

0 Permission entries

Outbound rules count

1 Permission entry

[Inbound rules](#)

[Outbound rules](#)

[Sharing](#)

[VPC associations](#)

[Tags](#)

>

EC2 > Load balancers > Create Application Load Balancer

Subnet

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale effectively.

subnet-0f267fd671d9f0c20

IPv4 subnet CIDR: 172.31.0.0/20

 ap-south-1c (aps1-az2)**Subnet**

Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale effectively.

subnet-0836785c0f40836ac

IPv4 subnet CIDR: 172.31.16.0/20

Security groups InfoA security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new one](#).**Security groups**

Select up to 5 security groups

default

sg-05dab4643acba95c0 VPC: vpc-03e487a9440b8ff6b

HTTP

sg-080dbce6964d40421 VPC: vpc-03e487a9440b8ff6b

>> Check the HTTP PORT and select the Target group

▼ Listener HTTP:80

Protocol

HTTP

Port

80

1-65535

Default action Info

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

Routing action Forward to target groups Redirect to URL Return fixed response**Forward to target group** InfoChoose a target group and specify routing weight or [create target group](#).**Target group**

target1

Target type: Instance, IPv4 | Target stickiness: Off

HTTP

Weight

1

Percent

100%

0-999

>> remaining leave every thing by default and create load balancer

<input type="checkbox"/>	Name	IP address type	VPC ID	Availability Zones	Security groups
<input type="checkbox"/>	applicationloadbalancer	IPv4	vpc-03e487a9440b8ff6b	3 Availability Zones	2 Security groups

INFO : 3 availability zone –load balancer it will distribute the traffic in three availability zone instances

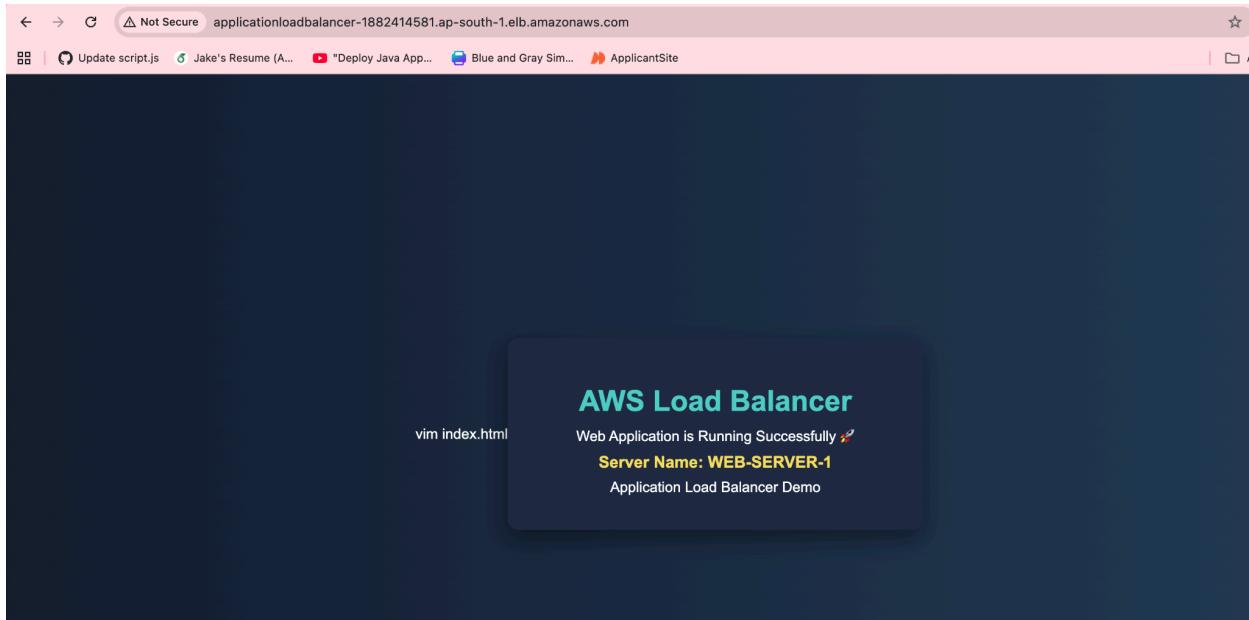
If the load state is in provisioning wait for few minutes them it will be in active

	Name	State	Type	Scheme	IP address type	
<input type="checkbox"/>	applicationloadbalancer	Active	application	Internet-facing	IPv4	

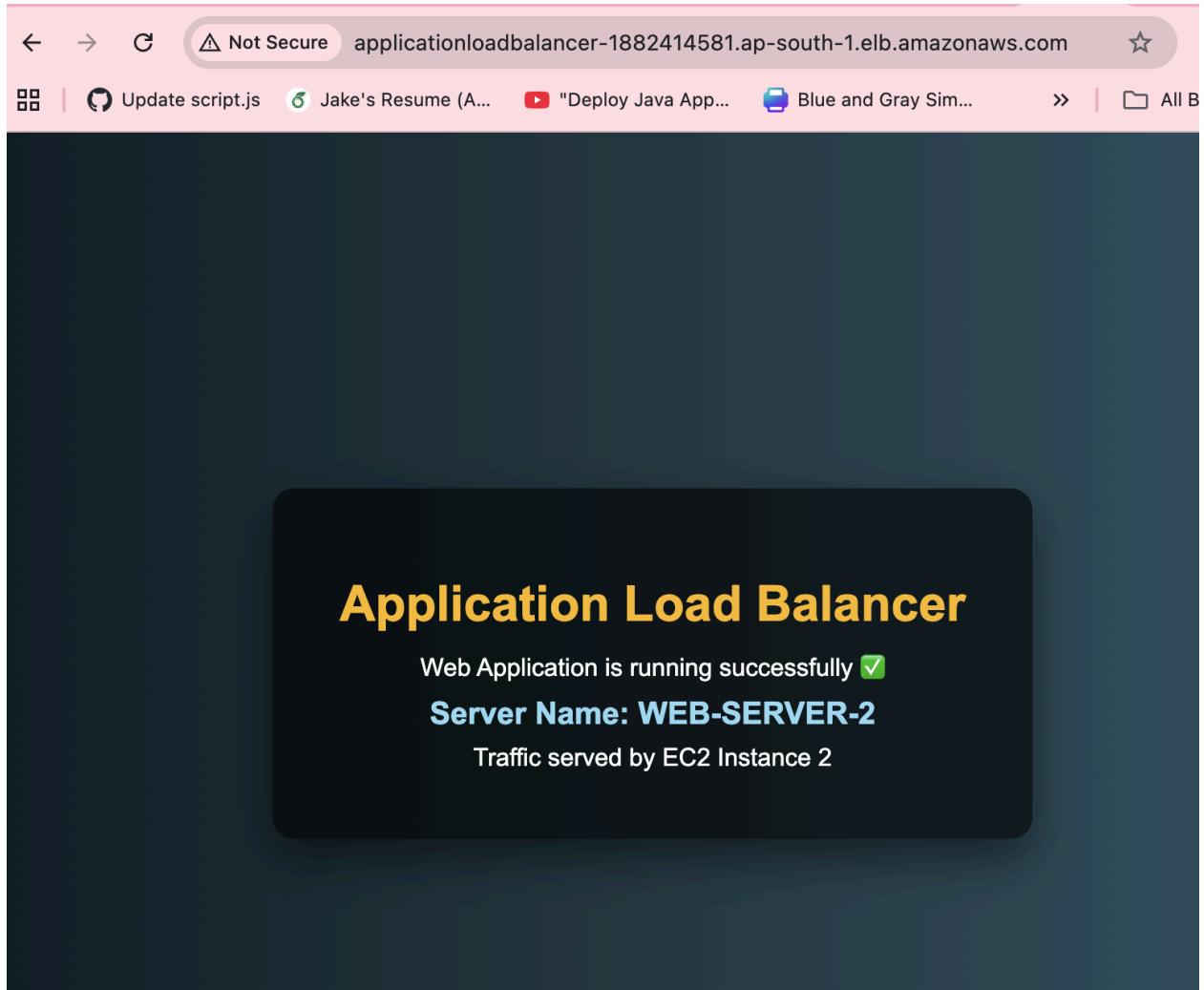
>To access the application no need to use the ip address use the DNS name for the safe and secure we can also send DNS name to the users (we are not using the ip address we only using the DNS name)

	Name	Availability Zones	Security groups	DNS name	ARN	Date created
<input type="checkbox"/>	applicationloadbalancer	3 Availability Zones	2 Security groups	<input type="text"/> applicationloadbalancer-18...	<input type="text"/> ARN	December 15, 2025

>> copy the DNS name paste it in chrome webserver-1 page will come



>> if we refresh the page webserver-2 page will come



>>INFO → traffic is distributing the by using the traffic group in load balancer. We can run multiple servers in traffic group. So it will distribute the traffic. When we send any request our target group will immediately take the action any server can give response to the user (like which ever server having low load the traffic group will send the request to that server)